Design and Development of Multirotor UAV with Automatic Collision Avoidance using Stereo Camera

Nicolas DEMIANNAY[1,2], Nicolas BESNIER[1,2], Cris THOMAS[3], Vikas THAPA[4], Vindhya DEVALLA[2], Amit Kumar MONDAL*[,5]

*Corresponding author
[1]École nationale supérieure d'ingénieurs de Caen, France,
nicodemi31@hotmail.fr, nicolas.besnier@icloud.com
[2]University of Petroleum and Energy Studies, Dehradun, Uttarakhand, India,
vindsdevalla@yahoo.com
[3]Università degli Studi di Genova, Genova, Italy,
cris.thomas1@gmail.com
[4]Indian Institute of Technology, Biomedical Optics Lab, Hyderabad, India,
vikas08thapa@gmail.com
*[,5]Manipal Academy of Higher Education,
Dubai International Academic City, Dubai, UAE,
akmondal1603@gmail.com

DOI: 10.13111/2066-8201.2020.12.3.5

Received: 14 May 2020/ Accepted: 28 July 2020/ Published: September (December) 2020

Abstract: *The operation of drones in cluttered environments like forests and hilly areas is extremely difficult; it is impossible to use drones autonomously without having built-in information to detect and avoid obstacles. The vision based obstacle avoidance algorithm is presented in this paper, with extensions to UAV navigation. The proposed method is incorporated on a stereo vison multi copter using a block matching algorithm. The stereo vision baseline is based on horizontal configuration and computes the depth using a sum of absolute difference algorithm. The image processing node (LabVIEW vi) and the controller node are run on a remote laptop. This vi computes the distance between the multirotor and an obstacle and transmits depth data to an onboard flight controller through the MAVLink protocol. The algorithm efficiency was tested using the software in the loop on Gazebo simulator to analyze the performance of the UAV. The hardware in loop results are also shown in this paper after the successful flight test.*

Key Words: *Robot Operating System (ROS), stereo vision, obstacle avoidance, LabView, Unmanned Aerial Vehicle (UAV), Gazebo, flight controller*

1. INTRODUCTION

Unmanned aerial vehicles (UAVs) have seen a great advance in recent years, due to their lower costs and increased productivity compared to conventional pilot aircraft as well as to their applicability in a wide range of fields. For example, the implementation of UAVs for the recognition and tracking of railways is safer and more efficient than the use of piloted, risk-oriented aircraft for the same purpose [1-7]. Unmanned aerial vehicles are also hugely successful in aerial photography, disaster management, and ground surveillance.

During the last decade, interest in UAVs and their autonomy has constantly increased [8]. The need for autonomy also arises sharply due to the skills and laborious work hours to be spent by UAV pilots at the ground station and this often compromises the prospectus in a variety of applications that UAVs encourage in the present and probable future. Despite the countless hours of training of the remote pilot before the actual flight, UAVs are still prone to accidents due to unprecedented obstacles or environments that which pilots cannot sometimes manage, and this results in huge losses both financially and in terms of corporate technology. Also, on the battlefield, UAVs can't dodge missiles without an adept pilot, which compromises military operations and makes mission vulnerable when the drone falls in enemy hands. All this brings the necessity of autonomous UAVs or drones to the forefront. Also on the battlefield, UAVs cannot avoid missiles without an adept pilot, which compromises military operations and makes the mission vulnerable when the drone falls into the hands of the enemy. All this brings to the fore the need for autonomous UAVs or drones.

Collision avoidance takes an important place for assisted flights. Although there are many solutions for obstacle avoidance, these solutions remain too expensive and most of the work is under laboratory scale. Only a few companies dominate the market for obstacle avoidance systems such as the Chinese DJI or the French Parrot.

This paper presents the obstacle avoidance using stereo vison in GPS denied environment for UAVs using robot operating system (ROS) which is open-source and acts as a meta-operating system for robots on Raspberry Pi and LabView on laptop. The image processing node and the controller node are run on a remote laptop, which receives real time images wirelessly. ROS installed on Raspberry Pi permits to read images from the stereo camera and transmits them to the laptop, also allowing to control the actuators through Pixhawk on UAV.

Limited autonomy of drones has already been achieved in terms of navigation from one point to another using GPS with limited accuracy of about 20 meters. Although the accuracy of GPS navigation is variable and regulated by US government to avoid abuse of it by third parties and hence cannot be relied upon for high accuracy [9-11]. Although navigation can be managed globally with GPS and locally, drones can have visual markers and other aids for precise location and landing, yet the biggest problem for autonomous drones during navigation to avoid collision or obstacles. It becomes increasingly difficult when the drone has to be flown in cluttered environments like forests and hilly areas even being manually piloted and it is impossible to be flown autonomously without some embedded intelligence for obstacle detection and avoidance. This has been a favorite research topic for unmanned ground vehicles (UGVs) for years and many techniques like optical flow, neural networks and nature inspired algorithms etc. based on textures and environment information learning etc., using sensors like ultrasonic sensors, monocular cameras [12-14], stereo cameras [15], LIDARs (Light Detection and Ranging) [16], laser range finders [17-19] etc., have been used [20]. But the same methods and techniques cannot be used for UAVs as they operate in three dimensions rather than two dimensions like ground vehicles. These sensors are expensive, heavy and consumes lot of power, which affects the endurance of an UAV. Additional features and modifications are to be done to apply the techniques of UGVs to UAVs for extracting and interpolation information in 3D for collision avoidance, where stereo camera proves to be much efficient.

The image coding plays a very important role in 3D information extraction in stereo vision. The motion of the object is determined using image coding. This translation movement generates frame to frame displacement of the moving objects [21]. Therefore, a displacement estimation is necessary for the UAV for identifying and avoiding the obstacle. The most popular way of estimating the translation motion is the block matching algorithm. Here the motion of the pixels (block) is represented using a displacement vector. This displacement vector is determined by the matching technique which generally follows three stages, namely motion detector to detect the moving blocks, displacement estimator to estimate the displacement vector and data compressor to encode the differences after motion compressor [22]. The choice of the appropriate algorithm depends on the target application and lot of models have been developed. The current video compression standards use a translational model with partitioned rectangular regions which transmits one motion vector to every region. These models are extensively used in video compression such as H.264 [23, 24] and MPEG [25]. The displacement estimator finds the best match of a reference block in a suitable match area. To do this, fast search algorithms are used to obtain the optimum point in a search region. Thus, the estimation problem results in a search problem. Most of these search problems focuses on the matching criteria with a reduced computational load without losing optimality [26]. To reduce resources and computation power required for the motion estimation, an efficient Sum of absolute differences (SAD) algorithm has been adopted as a matching criterion in this paper.

In stereo vision, the images that are captured using two cameras which are slightly displaced from each other. This positional difference is known as 'horizontal disparity' and gives rise to depth perception, even if the monocular images are unstructured and with noise and no distinctive features, as in random-dot stereograms [27]. The correspondence problem deals with the matching of monocular images by correlating the dots to the corresponding images. This 'correspondence problem' is a central issue that the visual system must solve to derive three-dimensional information.

The stereo correspondence problem is generally solved and confined to matching techniques, namely global and local. Local matching techniques consider small neighborhood of pixels. Block matching is one of the local matching techniques which proves to be much efficient compared to other techniques as discussed above and is most popularly used for stereo correspondence.

2. ALGORITHM

The Sum of Absolute Difference algorithm can be defined in the following equation:

$$I_{new}(x_0,y_o)=a_1I_{old}(x_1,y_1)+a_2I_{old}(x_2,y_2)+a_3I_{old}(x_3,y_3)+a_4I_{old}(x_4,y_4) \tag{1}$$

where I_{old} and I_{new} are the original and rectified image with the blending coefficients.

The SAD algorithm from the above equation shows an area-based correspondence algorithm [28-30].

g_t is the reference point of the left image and g_{t-1} is a point in epi-polar plane, it computes the intensity difference for each pixel

$$AD_v(x,y)=\sum j\sum i\,|g_t(x+i,y+j)-g_{t-1}(x+v+i,y+j)| \tag{2}$$

The above equation states as a reference point of the left image minus with the right image at the same epi-polar plane.

It sums up the intensities of all surrounding pixels in the neighborhood for each pixel of the left image. To calculate the stereo correspondence of the stereo images block, a matching technique is used.

Each block from the left image is matched into a block in right image by shifting the left block over the searching area of pixels in the right image. To find corresponding pairs of the stereo points, they must be compared for different disparities, after which the best matching pair can be determined. The maximum range at which the stereo vision can be used for detecting the obstacle depends on the image and the depth resolution. Absolute differences of pixel intensities are used in the algorithm to compute the stereo similarity between the points. By computing the SAD for pixels in a window surrounding the points, the difference between the similarity values for stereo points can be calculated. The disparity associated with the smallest SAD value is selected as best match [31].

Block matching algorithm or particularly feature based approach has been used to match the two images [32, 33].

It compares a reference block to next or previous block of pixels. If two different video from two different cameras are grabbed with just a different angle, a disparity matrix can be generated with a simple Block Matching algorithm.

From this matrix, the depth can be deduced with a correct calibration and transmitted to the embedded system.

The block-matching algorithm compares two 3 by 3 blocks of memory by accumulating either a sum of the absolute differences (SAD) between corresponding pixels or a squared sum of differences (SSD). The SAD algorithm has been used in this paper due to its quicker calculation, then a guileless usage of a relationship-based technique has been employed. The final summation is an accurate measure of how well two blocks of video match.

SAD method relates pixels from the two pictures are subtracted pairwise and the total distinction of their grey values is summed up.

As an example, a reference block is compared at a starting point, pixel by pixel, with the current block. On the target picture, three possibilities arise: left, right and center. On computing the difference between reference and target for these three cases depth can be obtained.

a) Algorithm on LabView

An algorithm to compute the depth image from the block matching algorithm is developed in LabView. NI Vision module along with ROS tool kit developed by Tufts University [34] were used to develop the algorithm. The video streams were received by HTTP stream in the form of MJPEG compressed by ROS.

Virtual web cams were simulated on laptop with VCAM software. VCAM software acts as a wireless interface between the LabView and the HTTP stream. All the inputs were fed in with respect to each camera in HTTP address. The VCAM software simulates the camera just like USB cameras, which can be accessed by NI Max. Therefore, NI Vision module can be accessed wirelessly.

The next step is to develop block matching algorithm in NI Vision module. The algorithm was divided into two categories:

a. to calibrate the camera and,
b. to compute the disparity and deduce depth from block matching algorithm.

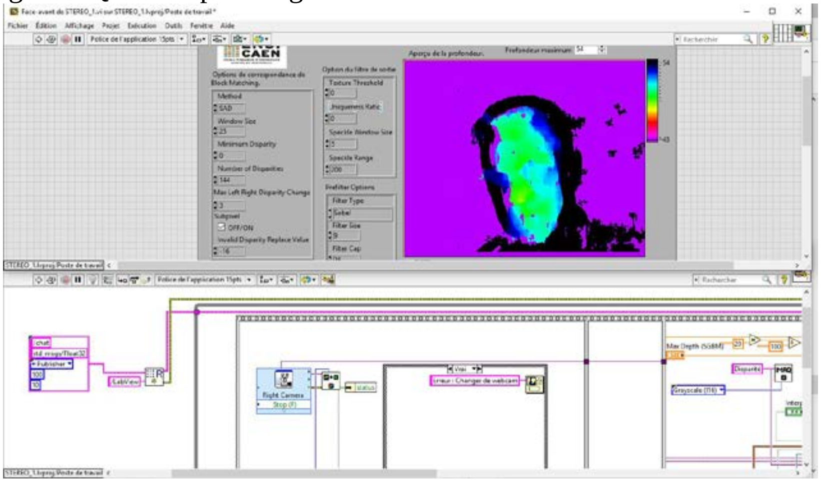
The calibration is necessary to interface the camera with LabView. Parameters like distance between the cameras, their orientation and angle of view are added and adjusted in calibration process.

The calibration is done by the visual interface (VI) developed by Mark Szaboo [35]. The calibration data file is obtained after the completion of the calibration.

The entire algorithm is divided into four steps:

a. In Express VI, cameras were run in 640x480 frames at 20 frames per second. The 32-bit color images are converted to 8-bit gray scale images to get depth image from stereo module. A binocular vision session is run in parallel. This binocular session reads the calibrated data. The output is two video streams and binocular session is calibrated. In fig. 1, step loop is used on a while loop has been used.

b. Block matching algorithm in NI Vision Module uses IMAQ stereo correspondence VI. This module needs four inputs and give two outputs. The calibrated *binocular stereo session*, *left image in* and *right image in* are the inputs and we get *disparity image Out* as the output. This module also takes in the *pre filter options*, *post filter options* and *correspondence options*. Thus the depth image is computed from disparity using IMAQ Get Depth image from Stereo VI module.



Options de correspondance de Block Matching.
Method: SAD
Window Size: 25
Minimum Disparity: 0
Number of Disparities: 144
Max Left Right Disparity Change: 3
Subpixel: OFF/ON
Invalid Disparity Replace Value: -16

Option du filtre de sortie
Texture Threshold: 0
Uniqueness Ratio: 0
Speckle Window Size: 5
Speckle Range: 200

Prefilter Options
Filter Type: Sobel
Filter Size: 9

Fig. 1 - LabView: Algorithm to Calculate the Depth

c. IMAQ Get Depth Image from Stereo VI module takes three inputs and gives three outputs. The inputs are calibrated binocular stereo session, disparity image and memory module to store the computed depth image. The binocular stereo session can be stopped whenever required and the depth image out can be recovered and converted into an array. Therefore, computed depth can be accesses in tabular form.

d. The last step shown in fig. 2 and 3 built a region of interest by computing mean of four depth data, which can be configurable according to the requirement. The final step is the communication between LabView and ROS. At the end of each iteration, LabView sends the depth data by publishing float32 with LabView node into chatter topic.

3. SYSTEM DESIGN

Two platforms were used to implement the algorithm: LabView on laptop and Robot Operating System (ROS) [36] on Raspberry Pi 3 [37]. ROS allows to link the camera to the hardware platform as well as the NI Vision module of LabView on the laptop. LabView student version has been used for the implementation of the project as it provides higher frame rates compared to OpenCV and Matlab Simulink tool (10 fps compared to 2 fps) [38], also allows interfacing of sensors and has NI Vision Module for image processing. Therefore two cameras of Microsoft studio lifecam with a resolution of 640×480 pixels at 20fps were used [39]. They are connected to Raspberry pi 3 using USB cables.

An image of Ubuntu mate 3 and ROS Kinetic was installed on Raspberry Pi 3. Ubuntu Mate 3 is a light version of Ubuntu for ARM processor as installed Raspberry Pi 3. ROS Kinetic version is chosen instead of the latest version because of its large community of developers on the Internet. ROS is a grouping of libraries, which permits to read data from the sensors and write the data on the computer /actuators/mechatronic system [40].

Cameras are connected to Raspberry Pi 3 and the system is mounted on the UAV, as shown in fig. 4.

Communication between raspberry pi 3 and laptop is done via Wi-Fi. The following chart in fig. 5, summaries the system architecture.

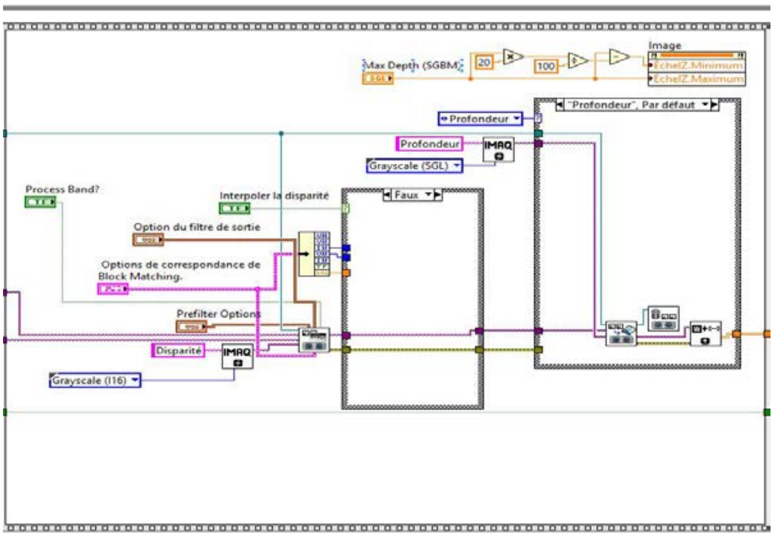


Fig. 2 - Computing part

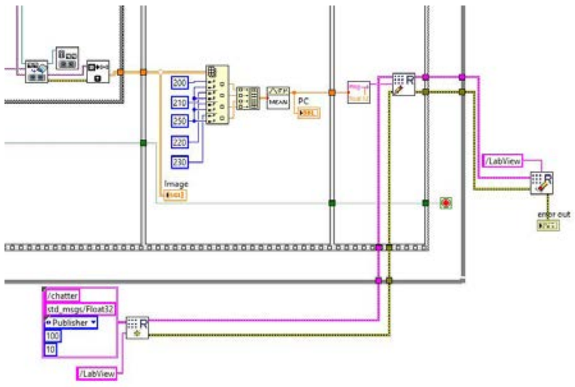


Fig. 3 - Communication between LabView and ROS

An image of Ubuntu mate 3 and ROS Kinetic was installed on Raspberry Pi 3. Ubuntu Mate 3 is a light version of Ubuntu for ARM processor as installed Raspberry Pi 3. ROS Kinetic version is chosen instead of the latest version because of its large community of developers on the Internet.

ROS is a grouping of libraries, which permits to read data from the sensors and write the data on the computer/ actuators/ mechatronic system [40].

Cameras are connected to Raspberry Pi 3 and the system is mounted on the UAV, as shown in fig. 4.

Communication between raspberry pi 3 and laptop is done via Wi-Fi. The following chart in fig. 5, summaries the system architecture.

It can deploy a network (in ROS) with the notion of node, topic, master, and message. Nodes can write messages into publisher topic and read messages into subscriber topic. A node can be a sensor (cameras in this case), a laptop or a C/ C++, Python or Java program. In the present case, wireless cameras with LabView has been used. Recover data from cameras and transmit computed data to a flight control board with just a Raspberry Pi 3.ROS operating scheme for the entire system has been shown in fig. 6.

Following ROS, packages were been used for cameras wired by USB on the Raspberry Pi 3:

- **uvc_camera**: This package allows cameras grabbing on raspberry pi.
- **mjpeg_server**: This package allows camera streams by HTTP.

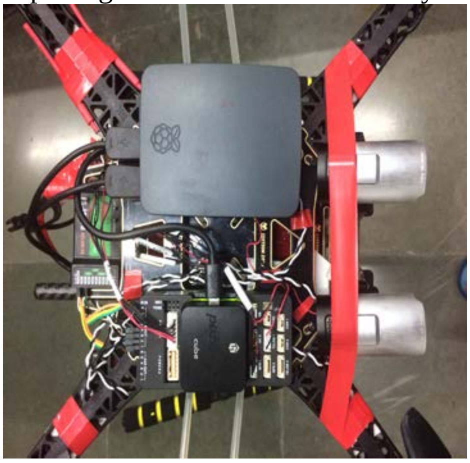


Fig. 4 - Assembling on UAV

There are four primary nodes. All are registered by ROS master.

- **Camera Node**: This node takes camera information and publishes it.
- **Web server Node**: This node takes camera information published by Camera Node and published it on the internet by the HTTP stream.
- **Image processingNode**: This node takes the HTTP stream and compute the depth by LabView software.
- **Flight controller Node**: This node takes the depth from the laptop node and takes the decision.

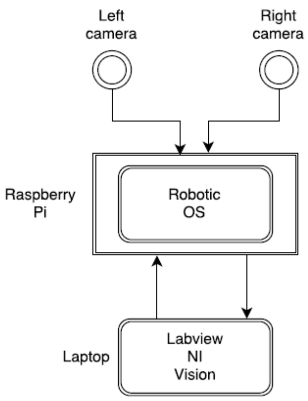


Fig. 5 - System architecture

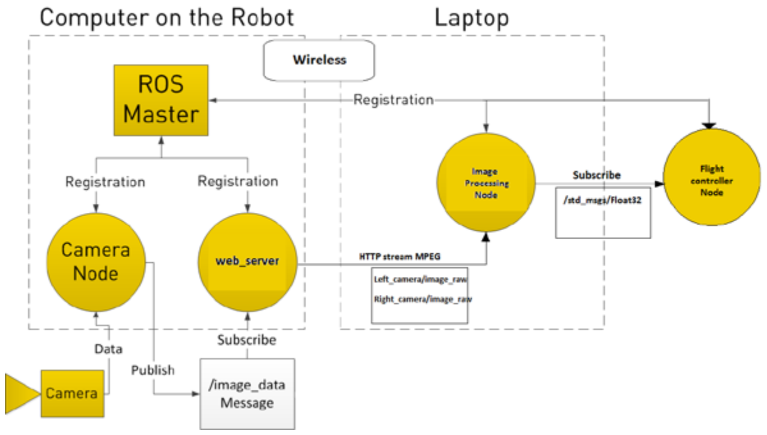


Fig. 6 - ROS operation scheme for the system

4. SIMULATION AND HARDWARE IMPLEMENTATION

An X-configuration quadcopter is used to implement the project. For the ground control station a QGroundControl (QGC) simulation software integrated with gazebo has been used [41]. The QGC software helps to make a backup of the parameters of the simulation and to implant them in the real system to approximate the conditions of the simulation. The simulation was done to test the functioning of the algorithm.

Both Hardware in the loop (HITL) and software in the loop (SITL) can be performed in the proposed simulation. SITL is used to check the efficiency of the algorithm in the software and HITL is used to check the efficiency of the algorithm in the hardware. To perform the SITL a similar quadcopter model was selected and “Commander Takeoff” command was written for the drone to takeoff.

The algorithm was then launched. It was observed that the drone makes a backward movement (as shown in fig. 7) when the obstacle was simulated using “Rostopic publish” towards the drone.

The algorithm included a command stating that if depth is equal to or less than 40cm. This was initiated by publishing “depth_topic”.

The simulation was useful to identify the values to be filled in topic “mavros/ actuator_control” to carry out the backward movement of drone. This algorithm proves to be good in GPS denied environment, i.e. mostly indoor.

As discussed earlier, using the QGC software, backup parameters were taken from the simulation to reuse them in the real hardware system so that the exact simulation can be recreated in the hardware.

a. *Flight Test*

The Success of the flight test depends on the implementation of the algorithm which collects depth data from ROS node. According to depth values, the UAV can turn back if an obstacle is approximately less than 40 cm and the UAV switches to off-board mode.

The off-board mode allows the computer to have the control on the UAV. To test this, a makeshift test stand with elastic band for safety tests was arranged, as shown in fig. 8.

The tests was not positive in the first attempt because the time between computing depth and the real turn back action was too slow. Despite that, the UAV was able to detect an object and make turns back, the performance can be visualized on YouTube platform [42] (Fig. 9)

5. DISCUSSIONS

The main problems are the latency due to the Wi-Fi signal and the delay due to depth computing. To solve this, it would be good to use a more powerful Wi-Fi transmitter. Moreover, to improve the computing depth, it would be good to use a more powerful processor.

To improve this work, a region of interest can be applied to avoid an obstacle on the left, the right, the bottom, and the top. For example, if UAV detects an obstacle on the right of the image or in the right region of interest, the UAV could be able to go to the left to avoid this obstacle. The same applies for the other directions.

The video transmission speed by the Raspberry Pi to the computer is not satisfactory, the Wi-Fi chip of the Raspberry Pi 3 and the computer card are not robust enough. However, this problem can be easily corrected by using Wi-Fi USB adapters. The system reacts slowly due to the hardware limitations of the computer. The result is that the number of frames per second is low, which considerably limits the speed of the drone.

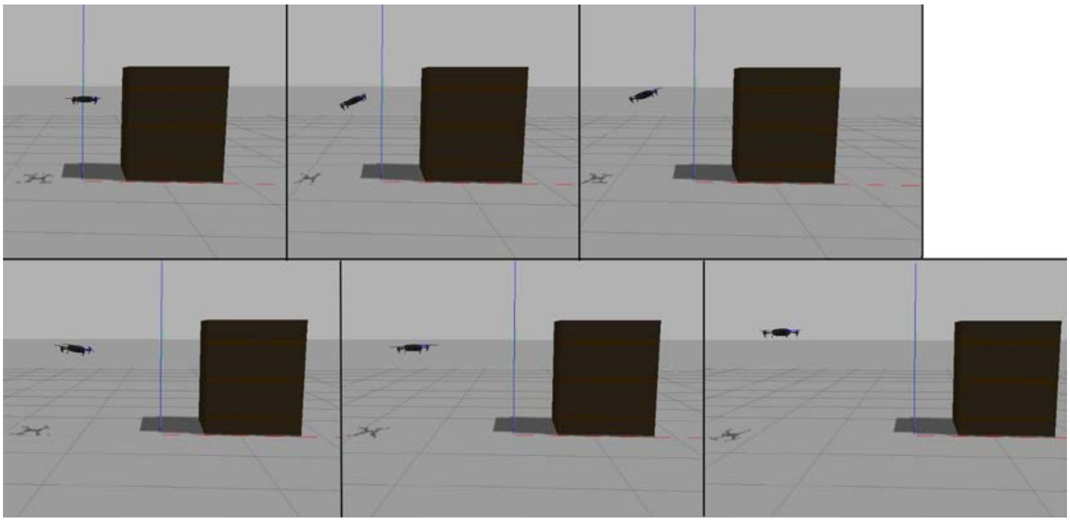


Fig. 7 - Simulation of the algorithm using Drone Iris+ from 3DR

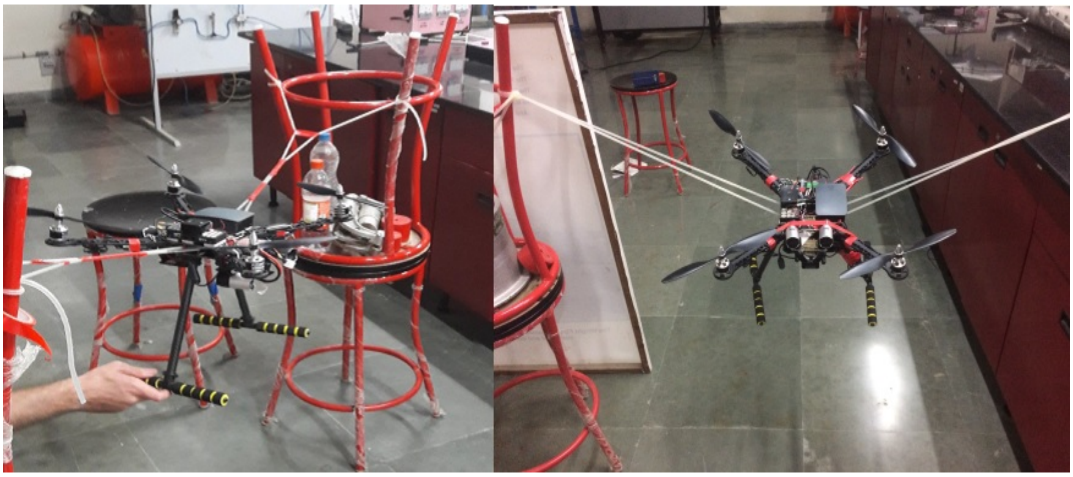
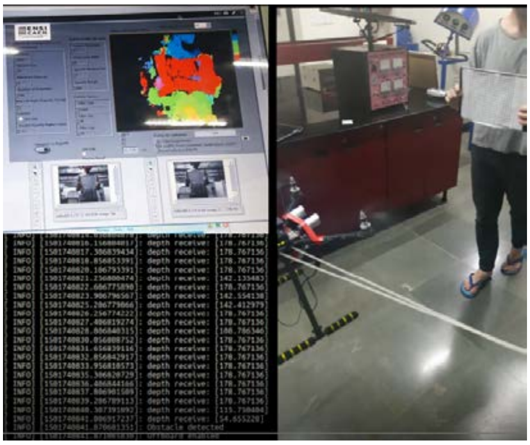


Fig. 8 - Test bed setup



```
INFO] [1501740816.156804017]: depth receive: [178.767136]
INFO] [1501740817.306839434]: depth receive: [178.767136]
INFO] [1501740818.856853391]: depth receive: [178.767136]
INFO] [1501740820.106793391]: depth receive: [178.767136]
INFO] [1501740821.256800474]: depth receive: [142.135483]
INFO] [1501740822.606775890]: depth receive: [178.767136]
INFO] [1501740823.906796567]: depth receive: [142.554138]
INFO] [1501740825.206779066]: depth receive: [142.412979]
INFO] [1501740826.256774222]: depth receive: [178.767136]
INFO] [1501740827.406819274]: depth receive: [178.767136]
INFO] [1501740828.806840315]: depth receive: [180.786346]
INFO] [1501740830.056808752]: depth receive: [178.767136]
INFO] [1501740831.656839116]: depth receive: [178.767136]
INFO] [1501740832.856842917]: depth receive: [178.767136]
INFO] [1501740833.956810573]: depth receive: [178.767136]
INFO] [1501740835.306820729]: depth receive: [178.767136]
INFO] [1501740836.406844166]: depth receive: [178.767136]
INFO] [1501740837.806901509]: depth receive: [178.767136]
INFO] [1501740839.206789113]: depth receive: [178.767136]
INFO] [1501740840.307391092]: depth receive: [115.750484]
INFO] [1501740841.006917237]: depth receive: [54.655228]
INFO] [1501740841.870601351]: Obstacle detected
```

Fig. 9 - Ongoing Test

6. CONCLUSIONS

To overcome the above discussed problems, the work can be extended on platforms like Nvidia Jetson board, BeagleBone board, as these systems allows to develop custom computer vision-based drones. These systems provide exceptional speed and power efficiency which is a major factor affecting the efficiency of the project.

REFERENCES

[1] F. G. Costa, J. Ueyama, T. Braun, G. Pessin, F. S. Osório, and P. A. Vargas, The use of unmanned aerial vehicles and wireless sensor network in agricultural applications, in *Geoscience and Remote Sensing Symposium (IGARSS), 2012 IEEE International*, pp. 5045-5048, 2012.

[2] V. Devalla, A. K. Mondal, A. A. J. Prakash, M. Prateek, and O. Prakash, Guidance, navigation and control of a powered parafoil aerial vehicle, *Current Science*, vol. **111**, p. 1045, 2016.

[3] V. Devalla, A. K. Mondal, and O. Prakash, Performance analysis of a powered parafoil unmanned aerial vehicle using open loop flight test results and analytical results, in *Research, Education and Development of Unmanned Aerial Systems (RED-UAS), 2015 Workshop on*, pp. 369-376, 2015.

[4] V. Devalla and P. Om, Longitudinal and Directional Control Modeling for a Small Powered Parafoil Aerial Vehicle, in *AIAA Atmospheric Flight Mechanics Conference*, p. 3544, 2016.

[5] V. Devalla, O. Prakash, and A. K. Mondal, Angle of Attack, Pitch Angle and Glide Angle Modeling at Various Thrust Inputs for a Powered Parachute Aerial Vehicle, in *Book of Abstracts*, p. 35, 2014.

[6] A. Ollero and L. Merino, Unmanned aerial vehicles as tools for forest-fire fighting, *Forest Ecology and Management*, vol. **234**, p. S263, 2006.

[7] Z. Sarris and S. Atlas, Survey of UAV applications in civil markets, in *IEEE Mediterranean Conference on Control and Automation*, p. 11, 2001.

[8] V. Devalla and O. Prakash, Developments in unmanned powered parachute aerial vehicle: A review, *IEEE Aerospace and Electronic Systems Magazine*, vol. **29**, pp. 6-20, 2014.

[9] A. Nemra and N. Aouf, Robust INS/GPS sensor fusion for UAV localization using SDRE nonlinear filtering, *IEEE Sensors Journal*, vol. **10**, pp. 789-798, 2010.

[10] H. Eisenbeiss, A mini unmanned aerial vehicle (UAV): system overview and image acquisition, *International Archives of Photogrammetry. Remote Sensing and Spatial Information Sciences*, vol. **36**, pp. 1-7, 2004.

[11] N. Abdelkrim, N. Aouf, A. Tsourdos, and B. White, Robust nonlinear filtering for INS/GPS UAV localization, in *Control and Automation, 2008 16th Mediterranean Conference on*, pp. 695-702, 2008.

[12] A. Kundu, K. M. Krishna, and C. Jawahar, Realtime multibody visual SLAM with a smoothly moving monocular camera, in *Computer Vision (ICCV), 2011 IEEE International Conference*, pp. 2080-2087, 2011.

[13] R. K. Namdev, A. Kundu, K. M. Krishna, and C. Jawahar, Motion segmentation of multiple objects from a freely moving monocular camera, in *Robotics and Automation (ICRA), 2012 IEEE International Conference on*, pp. 4092-4099, 2012.

[14] J. Chetan, K. M. Krishna, and C. Jawahar, An adaptive outdoor terrain classification methodology using monocular camera, in *Intelligent Robots and Systems (IROS), 2010 IEEE/RSJ International Conference on*, pp. 766-771, 2010.

[15] N. D. Reddy, I. Abbasnejad, S. Reddy, A. K. Mondal, and V. Devalla, Incremental real-time multibody VSLAM with trajectory optimization using stereo camera, in *Intelligent Robots and Systems (IROS), 2016 IEEE/RSJ International Conference on*, pp. 4505-4510, 2016.

[16] J. Liu, P. Jayakumar, J. L. Overholt, J. L. Stein, and T. Ersal, The role of model fidelity in model predictive control based hazard avoidance in unmanned ground vehicles using lidar sensors, *Ann Arbor*, vol. **1001**, p. 48109, 2013.

[17] C. Rasmussen, Combining laser range, color, and texture cues for autonomous road following, in *Robotics and Automation, 2002. Proceedings. ICRA'02. IEEE International Conference on*, pp. 4320-4325, 2002.

[18] B. M. Yamauchi, PackBot: A versatile platform for military robotics, in *Unmanned Ground Vehicle Technology VI*, pp. 228-238, 2004.

[19] M. Hebert, Active and passive range sensing for robotics, in *Robotics and Automation, 2000. Proceedings. ICRA'00. IEEE International Conference on*, pp. 102-110, 2000.

[20] M. H. Hebert, C. E. Thorpe and A. Stentz, *Intelligent unmanned ground vehicles: autonomous navigation research at Carnegie Mellon*, vol. **388**: Springer Science & Business Media, 2012.

[21] H. G. Musmann, P. Pirsch, and H.-J. Grallert, Advances in picture coding, *Proceedings of the IEEE*, vol. **73**, pp. 523-548, 1985.

[22] A. Puri, H.-M. Hang, and D. Schilling, An efficient block-matching algorithm for motion-compensated coding, in *Acoustics, Speech, and Signal Processing, IEEE International Conference on ICASSP'87*, pp. 1063-1066, 1987.

[23] H. Schwarz, D. Marpe, and T. Wiegand, Overview of the scalable video coding extension of the H. 264/AVC standard, *IEEE Transactions on circuits and systems for video technology*, vol. **17**, pp. 1103-1120, 2007.

[24] T. Wiegand, G. J. Sullivan, G. Bjontegaard, and A. Luthra, Overview of the H. 264/AVC video coding standard, *IEEE Transactions on circuits and systems for video technology*, vol. **13**, pp. 560-576, 2003.

[25] D. Le Gall, MPEG: A video compression standard for multimedia applications, *Communications of the ACM*, vol. **34**, pp. 46-58, 1991.

[26] M. Brunig and W. Niehsen, "Fast full-search block matching," *IEEE Transactions on circuits and systems for video technology*, vol. **11**, pp. 241-247, 2001.

[27] A. Nieder, Stereoscopic vision: Solving the correspondence problem, *Current Biol.*, vol. **13**, pp. R394-R396, 2003.

[28] A. Fusiello, E. Trucco, and A. Verri, A compact algorithm for rectification of stereo pairs, *Machine Vision and Applications*, vol. **12**, pp. 16-22, 2000.

[29] L. Di Stefano and S. Mattoccia, Fast Stereo Matching for the VIDET System using a General Purpose Processor with Multimedia Extensions, in *camp*, pp. 356-, 2000.

[30] A. Kuhl, Comparison of stereo matching algorithms for mobile robots, *Centre for Intelligent Information Processing System*, pp. 4-24, 2005.

[31] J. C. van den Heuvel, J. Kleijweg, W. van der Mark, M. Lievers, and L. Kester, *Obstacle detection for people movers using vision and radar*, 2003.

[32] A. Howard, Real-time stereo visual odometry for autonomous ground vehicles, in *Intelligent Robots and Systems, 2008. IROS 2008. IEEE/RSJ International Conference on*, pp. 3946-3952, 2008.

[33] Y. Jiang, Y. Xu, and Y. Liu, Performance evaluation of feature detection and matching in stereo visual odometry, *Neurocomputing*, vol. **120**, pp. 380-390, 2013.

[34] * * * T. U. b. t. M. E. D. a. t. C. f. E. E. a. Outreach, *ROS for LabView Software*, ed: GitHub, 2017.

[35] M. Szabó. (2016). *Machine Vision - Image processing and depth image with two webcams*, Available: https://1drv.ms/

[36] * * * O. S. R. Foundation. (2018, 1/1/2018). *Robotic Operating System*. Available: http://www.ros.org/

[37] * * * R. P. Foundation. (2017, 31/12/2017). *Raspberry Pi 3 Model B*, Available: https://www.raspberrypi.org/products/raspberry-pi-3-model-b/

[38] H. Kodam, *Quad rotor Based Surveying and Tracking Tool to be applied in the Agricultural Industry*, 2013.

[39] * * * Microsoft. (2017). *LifeCam Strudio*, Available: https://www.microsoft.com/accessories/en-us/products/webcams/lifecam-studio/q2f-00013

[40] * ** (O. S. R. Foundation. (2017, 1/1/2018). *Why ROS*, Available: http://www.ros.org/core-components/

[41] * * * Q. D. Control. (2017). *QGroundControl*, Available: http://qgroundcontrol.com/

[42] D. Nicolas. (2017). *UAV obstacle avoidance ROS*, Available: https://www.youtube.com/watch?v=IaUEVV9D40k